

REMARKS

Status of this application

In the Office Action mailed on May 4, 2006, Claims 1-16 were rejected under 35 U.S.C. 102(e) as being anticipated by Nesbitt U.S. Application Publication No. 2004/0042405 (hereinafter "Nesbitt"). Claims 1, 2, 6-16 were further rejected under 35 U.S.C. 102(e) as being anticipated by Nevin III Patent 6,714,936 (hereinafter "Nevin").

This amendment amends the Abstract to correct typographical error and also amends claim 4 to correct an error and clarify its meaning.

Reconsideration of the rejections of the claims based on the teachings of Nesbitt and Nevin is requested for the reasons detailed below. The remarks that follow are organized in the same order that the rejections were presented in the outstanding action.

The rejections based on Nesbitt

Claims 1-16 were rejected under as being anticipated by Nesbitt. Reconsideration is requested since Nesbitt does not disclose or suggest the subject matter set forth in the rejected claims.

Claim 1 is directed to a relational database management system (RDBMS) that employs two relational tables with specifically described content and which includes an application program interface that enables executing application programs to perform specifically described functions using those tables. Nesbitt does not mention relational database management systems, does not describe providing an API of any kind, does not describe the specific tables set forth in claim 1, and does not describe the specific functions which the API permits executing applications to perform with respect to the data in those tables.

Nesbitt describes a mechanism for determining a preferred route from a point of origin to a destination in a network of nodes connected by links. Nesbitt analyzes data which defines the network, but does not teach a relational database system that permits executing application programs to create tables in a RDBMS, to store data in these tables, or to perform standard operations on those tables. The advantages which are provide by implementing standard network analysis functions in an RDBMS which permits executing application programs to create and use RDBMS tables that can also be manipulated by standard RDBMS are detailed in

applicant's "Summary of the Invention" at pages 3-4 of applicant's specification. Nesbitt nowhere discloses or suggests this general architecture which is clearly required by claim 1.

As the Examiner points out, Nesbitt employs node information at 850 and link information at 855. The node information 850 consists of a set of records each of which is shown in detail at 600 in Fig. 6 and a node identifier 610 as well as information about the links at 620, 630 and 640 (see [0093]). The link information 855 consists of a set of records each of which is shown in detail at 700 in Fig. 7 and described at paragraphs [0094]-[0095].

While similar to applicants' claimed generic node table and generic link table, the data structures which Nesbitt describes differ in important ways from the node table and link table claimed by applicant. In Nesbitt's data structure, the records describing nodes seen in Fig. 6 include a node identification as well as an identification of the link(s) which connect to that node, whereas the link information at 700 describes a link but identifies only the end node. This works well for route determination where the data need only identify the links which are connected to a given node from which the route will proceed, but would complicate the performance of many of the standard operations which are made available to executing program's by applicant's API, such as "storing data describing links between nodes" (which in applicant's system would simply add rows to the link table but which, in Nesbitt's system, would require adding a row to the link table and posting further data into the list of links in the node information structure), and performing standard operations on the link table as claimed in claim 1. Applicant has amended claim 1 to clarify the fact that link table contains rows which identify both nodes connected by each link, rather than only the end node as done in Nesbitt's link information record 700.

At paragraph [0042], Nesbitt indicates that his system uses a "directed link list" that may be "organized as a data set, a database table, an object instance, a linked list, a XML-document, a text file, or another data structure." And at paragraph [0045], Nesbitt states that "To identify a directed link that is adjacent to the end node, the routing system may access node information stored in a table or list. The node information may include a list of the directed links that are adjacent to each node in the routing graph." But the lists referred to do not satisfy the express definitions of the "generic node table" or the "generic link table" as set forth in claim 1.

The Examiner suggests that Nesbitt teaches a generic node table at 850 and in Figs. 3, 5 and 8. The only references to the "node information 850" are found in paragraphs [0098] which states that node information 850 is one of the data components and paragraph [0100] which states that "The route data processor 835 may use geographic, road or other routing data to transform the routing data into node information 850 and directed link information 855 that may be used in determining a preferred route." Node information 855 (sic – probably 850 was intended) may include information about particular nodes, such as the node information described with respect to FIG. 4 and the node information data structure 600 in FIG. 6.

Claim 1 further expressly requires the presence of an application program interface which enables executing application programs to:

- create said node table and said link table,
- to store data describing nodes in said node table,
- to store data describing links between said nodes in said link table, and
- to perform a plurality of standard operations on the data in said node table and said link table.

Nesbitt does not describe an API that enables executing application programs to do anything. Nesbitt suggests at [0100] that the "route data processor 835 may use geographic, road or other routing data to transform the routing data into node information 850 and directed link information 855 that may be used in determining a preferred route" but there is no suggestion that Nesbitt's node and link information is created by using the API of an RDBMS to create tables, store data in tables, or perform standard operations on the data in these tables as expressly required by claim 1.

Reconsideration of the rejection of independent claim 1 as being anticipated by Nesbitt is accordingly requested. The remaining claims 2-16 are all dependent on claim 1 and are hence also allowable. In addition, these dependent claims set forth additional features and functions that are not described by Nesbitt as noted below.

Claim 2 states that the network is "logical network." Applicants' specification, at [041] defines a "logical network" as follows: "A logical network contains connectivity information but no geometric information. This is the model used for network analysis. A logical network

can be treated as a directed graph or undirected graph, depending on the application.” Nesbitt’s system can be used with “logical networks” and hence applicants’ rely on the limitations in parent claim 1 which distinguish over Nesbitt.

Claim 3 requires that the rows of both the generic node table and the generic link table include a cost attribute. The description of the node record seen in Fig. 6 does not include a node cost attribute. Nesbitt does teach that costs are associated with links and intersections at [0035]. While “intersection costs” are discussed at [0095] and are stored with the link information 700 at 760 (not in a row of the node table as required by claim 3), there is no explanation of where “link costs” are stored, and intersection costs are the only costs shown the link information 700. Thus, Nesbitt does not teach that link costs are stored in link tables, and does not teach that node costs (intersection costs) are stored in the node tables (but are instead stored with the link information). Claim 3 accordingly distinguishes over the Nesbitt teaching for this additional reason.

Claim 4 is dependent on claim 3 and further requires that one of the standard operations that an application program can perform using the RDBMS API is to identify a particular path having one or more stated characteristics. Nesbitt’s routing methods can identify paths having the lowest cost as noted by the Examiner, but these methods are not performed by using an API to perform operations on the node table and the link table as claimed.

Claim 5 is also dependent on claim 3 and further specifically requires that one of the standard operations is the identification of the least cost path from a stated start to end node in the network. As noted with respect to claim 4, Nesbitt performs this function, but does not do so using an RDBMS API as claimed.

Claim 6 requires that one of the standard operations is the identification of a path consisting of an alternating sequence of nodes and links having defined characteristics. Nesbitt’s route finder performs that function, but does not do so using an RDBMS API as claimed.

Claim 7 requires that the RDBMS of claim further includes a path table containing a plurality of path table rows each of which contains data describing a path consisting of an alternating sequence of nodes and links. An example is seen in applicant’s Fig. 4. The Examiner cites Fig. 5 which shows the results obtained when Nesbitt calculates a preferred route. The information structure seen in Nesbitt’s Fig. 5 does describe a path and contains rows, but each row does not describe a path. Instead, each row in Fig. 5 describes a link, and it requires a

plurality of rows to describe a path. Thus, while Nesbitt does store data describing paths, it does not do so in the way required by claim 7.

Claim 8 is dependent on claim 7 and requires that an RDBMS API standard operation is used to identify a path and place information describing that path in one row of the path table. Nesbitt does not perform RDBMS API operations and does not describe placing path descriptions in a single row of a path table.

Claim 9 is also dependent on claim 7 and requires the creation of a path-link table containing one ordered set of path-link table rows associated with each given path described in said path table, each of said path table rows containing information identifying one link in the sequence of links in said given path. Nesbitt creates a path-link table as seen in Fig. 5 but this table does not describe a path in a path table.

Claim 10 is dependent on claim 9 and requires that an RDBMS API standard operation is used to identify a path and place information describing that path in one of the path table rows and also placing a sequence of links in a path link table. Nesbitt does not perform RDBMS API operations of any kind, nor does it perform the other stated functions for the reasons noted above.

Claim 11 states the standard RDBMS API operations includes loading node and link data into the node and link tables respectively from a database. The cited paragraphs 0020, 0021, 0024, and 0028 do not describe this operation. The closest reference is the statement in 0028 that typical client computer systems use application programs, one of which may be a database program. That does not suggest that network analysis system should be implemented as an RDBMS, should have an API that permits executing applications to perform standard operations on the network data, and that one of these standard operations should be loading network data from a database.

Claim 12 states that the network is a "spatial network." As defined in applicants' specification at [042], a "spatial network" contains both connectivity information and geometric information. The manner in which applicants' RDBMS network model stores geometric information is described in detail beginning at [090] in applicant's specification. The Examiner suggests that Nesbit describes storing both connectivity information and geometric information, citing Figures 3 and 3A and paragraph 20. Nothing in those cited portions of Nesbit describe any data that would define the shape of any node as required by claim 12.

Claim 13 is dependent on claim 12 and further requires that each link table a column for storing the identification of a geometry object which specifies the geometry of one of said links. The closest thing to a link table row is the link information 700 (Fig. 7) which does not specify a geometry object of any kind, and none of the cited sections (Figures 3 and 3A and paragraph 20) of Nesbitt do either.

Claim 14 requires that each of the link table rows includes a column for storing the identification of a geometry object which specifies the shape and location of one of said links. Nesbitt does not disclose geometry objects which specify the shape and location of links anywhere.

Claim 15 requires that each of said node table rows further contains a level column for holding a hierarchy level. Nesbitt nowhere mentions hierarchies or hierarchy levels, and the node information record seen in Fig. 6 contains no hierarchical. The Examiner suggests that "Nesbitt discloses that the each of said node table rows further contains a level column for holding a hierarchy level (Figure 1 is the parent link and Figure 3 is links) (See Figures 1 and 3)." This observation is not understood. Fig. 1 is a block diagram of the system and Fig. 3 shows a network graph, but neither shows "parent links."

Claim 16 is dependent on claim 15 and further requires that each of said node table row further contains a parent column for holding the identification of a parent node within the hierarchy established by said level column. The Examiner suggests that "Nesbitt discloses that each of said node table rows further contains a parent column (routing graph) for holding the identification of a parent node (See paragraph 8) within the hierarchy established by said level column." Again, this suggestion is not understood. Paragraph 8 states no-outlet link may be identified in a routing graph and the node information structure seen in Fig. 6 includes a list of links from that node. But these links connect nodes in a directed graph and are nowhere suggested to establish hierarchies having levels and there is no suggestion that levels in a hierarchy are specified anywhere nor is there any suggestion that the node information specifies not only levels but also the parent node of any level. The node information in Fig. 6 does not identify any other node for any purpose.

Claims 1-16 are accordingly believed to be patentable over Nesbitt for the reasons given above.

The rejections based on Nevin

Claims 1, 2 and 6-16 were rejected as being anticipated by Nevin. Reconsideration is requested for the reasons given below.

Claim 1 is directed to a relational database management system (RDBMS) that employs two relational tables with specifically described content and which includes an application program interface that enables executing application programs to perform specifically described functions using those tables. Nevin, in contrast, does not use an RDBMS and indeed teaches away using such a system. Nevin calls his database system "DataSea" and states at col. 2, lines 7-13 that *"DataSea can serve as the single source of data for any application. Any RDBMS (relational database management system) can do this, but DataSea is completely flexible in its data storage and linkage, guaranteeing forward compatibility by eliminating the risk of changes to database structure and entity-relationships of RDBs (Relational Databases)." And again at col. 8, lines 35-38: "While DataSea can emulate a RDBMS, without the complications of tables and foreign keys, the rich connections of DataSea and its ability to insert abstract nodes opens the way for neural-type processing." And further at col. 13, lines 31-33: "The user need not know about the data structure, such as database tables and their entity relationships in a relational database, or the directory structure of a file system". Nevin's DataSea system does provide an API, but it does create, store or perform operations on relational tables as claimed. Nevin states that "The DataSea API (application programming interface) provides access to all data while eliminating the need to worry about database structure such as tables, columns and foreign keys." (Col. 2, lines 38-41).*

Nevin does not store data in either a generic node table or a generic link table as claimed. The Examiner cites Nevin at See Col. 17, lines 1-39 which states expressly that Nevin's database *"differs in significant ways from RDBMS's".* Nodes, as Nevin uses that terms, are not nodes in a network joined by links to form relationships between nodes as claimed, but instead repositories of data that can be linked together, including web pages, or objects as complex as a virtual reality view of a manufacturing facility. There is no suggestion in the cited passage of "a generic node table (in an RDBMS) containing a plurality of node table rows each of which contains data describing a given node in a network" as claimed. Nor is there any teaching in that cited passage of "a generic link table containing a plurality of link table rows each of which contains data

describing a link between two nodes in said network and identifying each of said two nodes” as required by claim 1.

Nevin nowhere suggests that an API is used to create, store or perform operations on relational tables as claimed. The cited passage at col. 13, lines 40-46 suggests that DataSea has an API but does not suggest that DataSea ever creates database tables as recited in claim 1. The cited abstract and Col. 13, lines 20-30 do not suggest that data describing nodes is stored in a node table, or that data describing links is stored in a link table, by the DataSea API or anything else. The citation at col. 13, lines 20-30 describes how RDBMSs work, but does not suggest that DataSea does this, and does not suggest that RDBMS systems generally store node and link tables of the type claimed. None of the passages at Col. 7, lines 34-43, Col. 13, lines 41-46 and Col. 14, lines 64-67 describes an RDBMS API that performs any function, let alone the specific standard operations on node and link tables recited in claim 1.

Reconsideration of the rejection of claim 1, and its dependent claims 2 and 6-16, is accordingly requested. The dependent claims rejected based on Nevin are further allowable for the reasons stated below:

Claim 2 states that the network is “logical network.” Applicants’ specification, at [041] defines a “logical network” as follows: “A logical network” contains connectivity information but no geometric information. This is the model used for network analysis. A logical network can be treated as a directed graph or undirected graph, depending on the application.” Nevin’s system, as pointed out above, does not analyze “network data stored in relational tables that describe a set of nodes and links forming a network.” Nevin’s “nodes” are data storage objects that are linked together, but there is no suggestion that Nevin’s system is used for network analysis or that the data which Nevin stores in his “DataSea” of linked nodes describes “a set of nodes and links forming a network.” as claimed.

Claim 6 requires that one of the standard operations performed by the RDBMS API is the identification of a path consisting of an alternating sequence of nodes and links having defined characteristics. The Examiner cites Nevin at Col. 5, lines 17-18, Col. 17, lines 129 and Col. 24, lines 34-59). These passages have been carefully reviewed and none were found to teach the operation set forth in claim 6, and certainly not by an RDBMS API standard operation as claimed.

Claim 7 requires that the RDBMS of claim further includes a path table containing a plurality of path table rows each of which contains data describing a path consisting of an alternating sequence of nodes and links. An example is seen in applicant's Fig. 4. The Examiner cites Nevin at Col. 17, lines 1-29, but that passage expressly states that relational tables are not employed and nowhere suggests that comparable data is placed anywhere else.

Claim 8 is dependent on claim 7 and requires that an RDBMS API standard operation is used to identify a path and place information describing that path in one row of the path table. The Examiner cites Nevin at Col. 10, lines 1-34, Col. 16, lines 59-68, Col. 17, lines 1-29. Each of these passages has been reviewed and none have been found to disclose or suggest path tables containing information describing a path each row of a path table as claimed. If anything, these passages again teach away from the use of RDBMS tables.

Claim 9 is also dependent on claim 7 and requires the creation of a path-link table containing one ordered set of path-link table rows associated with each given path described in said path table, each of said path table rows containing information identifying one link in the sequence of links in said given path. The Examiner cites Nevin at Col. 5, lines 17-18, Col. 10, lines 1-34, Col. 16, lines 59-68, and Col. 17, lines 1-29. Nevin does establish paths through linked data nodes for various purposes, but there is no suggestion in any of these passages that anything comparable to the claimed path link table containing an ordered set of path link table rows is ever created as claimed.

Claim 10 is dependent on claim 9 and requires that an RDBMS API standard operation is used to identify a path and place information describing that path in one of the path table rows and also placing a sequence of links in a path link table. The Examiner cites the same passages relied upon above with respect to claim 9, but none of these describe an RDBMS API operation of any kind and no operation of any kind performed by Nevin stores or processes the tables as set forth in claim 9.

Claim 11 states the standard RDBMS API operations include loading node and link data into the node and link tables respectively from a database. Nevin does not describe RDBMS API operations, and does not describe node or link tables. The cited passages at Col. 7, lines 34-43, Col. 13, lines 41-46, Col. 14, lines 64-67, Col. 16, lines 59-68 and Col. 23, lines 54-65 have been reviewed and none were found to disclose the subject matter of claim 11.

Claim 12 states that the network is a "spatial network." As defined in applicants'

specification at [042], a "spatial network" contains both connectivity information and geometric information. The manner in which applicants' RDBMS network model stores geometric information is described in detail beginning at [090] in applicant's specification. The Examiner suggests that Nevin discloses this claimed subject matter at Col. 7, lines 34-42, Col. 13, lines 41-46, Col. 15, lines 5-38, Col. 16, lines 59-68 and Figures 1, 4-9. As noted above, Nevin describes data nodes that have links, but does not describe a system for analyzing network data stored in relational tables (or anywhere else) that describe a set of nodes and links forming a network. None of the cited passages discloses or suggests that the "shape and location of "nodes in such a network are described in a column in a row of a node table as claimed.

Claim 13 is dependent on claim 12 and further requires that each link table a column for storing the identification of a geometry object which specifies the geometry of one of said links. The cited passages at Col. 7, lines 34-42, Col. 13, lines 41-46, Col. 15, lines 5-38, and Col. 16, lines 59-68, and Figures 1, 4-9, nowhere describe links that have geometry. In Nevin, links are links between data storage objects, not links in a directed graph network, and they don't have a "geometry."

Claim 14 requires that each of the link table rows includes a column for storing the identification of a geometry object which specifies the shape and location of one of said links. The Examiner cites Col. 7, lines 34-42, Col. 13, lines 41-46, Col. 15, lines 5-38, Col. 16, lines 59-68 and Figures 1, 4-9, but as noted earlier, neither these passages nor anything else in Nevis suggests that links have shapes and locations, let alone that these shapes and locations are specified by geometric objects stored in a column of a link table as claimed.

Claims 15 and 16 requires that each of said node table rows further contains a level column for holding a hierarchy level and that each of said node table row further contains a parent column for holding the identification of a parent node within the hierarchy established by said level column. Nevin notes in several places that his system permits data to be visualized in hierarchical levels, but nowhere suggests that data about hierarchical levels is stored in relational tables as claimed; indeed, Nevin makes it a point to frequently observe that he does not use relational tables at all but rather stores data in linked data objects he calls nodes.

Claims 1-16 are accordingly believed to be patentable over Nevin for the reasons given above.

Conclusion

It is submitted that neither Nesbitt nor Nevin disclose applicants' invention as claimed. While Nesbitt is concerned with performing the analysis of networks of nodes and links to find the lowest cost route between a starting and ending point, Nevin is concerned with using a linked data object database to better visualize data and is not concerned with analyzing networks of data in directed graphs.

Neither Nesbitt nor Nevin describe a relational database management system which provides an API to executing applications to permit them to create, store and manipulate node and link tables. Nesbitt says nothing about RDBMS systems or APIs, and Nevin teaches that RDBMS systems are poor choices for performing the tasks Nevin seeks to accomplish.

Claims 1-16 describe in detail numerous specific table data structures and data manipulation operations on those structures that are plainly not taught by either of the cited references.

This application as now presented is accordingly believed to be in condition for allowance.

Respectfully submitted,



Dated: October 4, 2006

Charles G. Call, Reg. 20,406

Certificate of Transmission under 37 CFR 1.8

I hereby certify that this *Amendment* is being transmitted by facsimile to the central facsimile number of the U.S. Patent and Trademark Office, (571) 273-8300, on October 4, 2006.



Dated: October 4, 2006

Signature _____

Charles G. Call, Reg. No. 20,406
USPTO Customer No. 021253
68 Horse Pond Road
West Yarmouth, MA 02673
Ph. (508) 778-2630 Fax (508) 629-6540
call@patentsoft.com